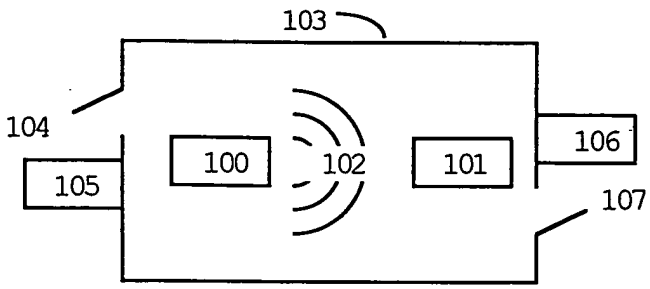

FIG. 1

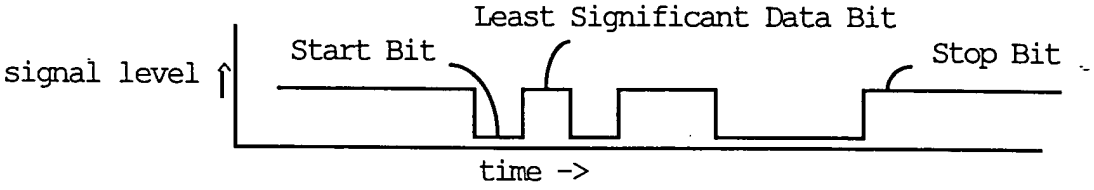

FIG. 2

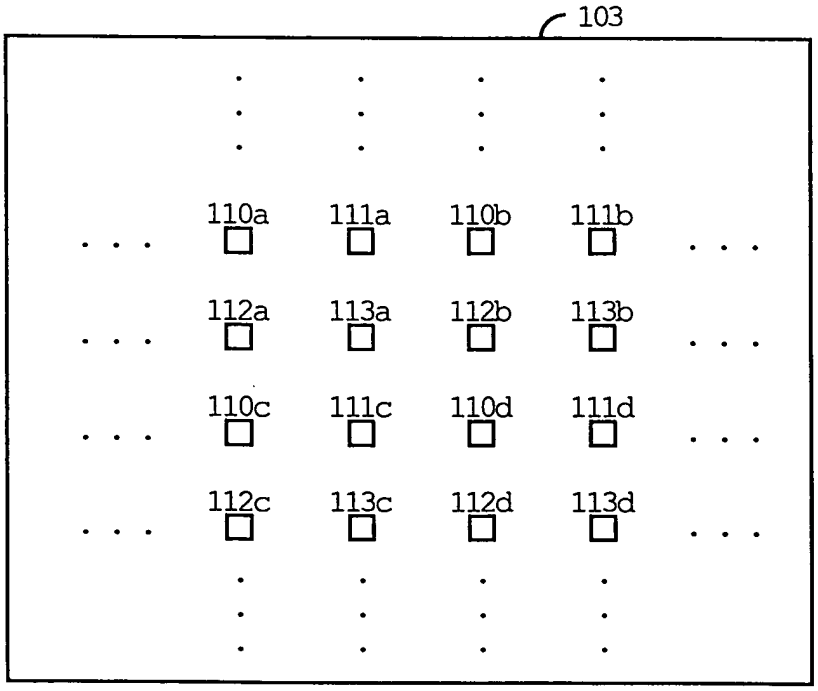

FIG. 3

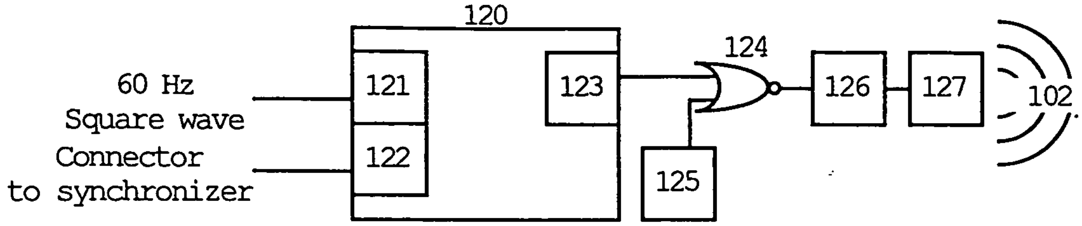

FIG. 4

```
volatile unsigned short EPDDR@0x10007002, EPDR@0x10007004, U0RX@0x10009000,
 U0TX@0x10009040, U0CR1@0x10009080, U0CR2@0x10009082,
 U0BRGR@0x10009084, U0SR@0x10009086, U0PCR@0x1000908A,
 EPPAR@0x10007000, EPFR@0x10007006;                          129

enum{ UART_EN = 1, TXEN = 0x1000, RXEN = 0x100, IRTS = 0x4000, WS = 0x20,
 TRDY = 0x2000, RRDY = 0x100};                               130
```

FIG. 5

```
                        131
void init() {                                               132
 U0BRGR=3333; /* 600 baud if system clock is 32 MHz */
  U0PCR = 2; // Txd pins used by UART0
 U0CR1 = UART_EN + TXEN; // enable UART transmitter
 U0CR2 = IRTS + WS; // disable hardware RTS control, have 8-bit data frame
 EPPAR = 5; // make both inputs rising edge sensitive
}                                                               133
                                          134
void put() { short i; static char count;      135
  if(((i = EPFR) & 1) == 0) return;              136
 EPFR = i; if(i & 2) count = 0;                              137
 if((((count++) & 3) == 0) && (U0SR & TRDY)) U0TX = 0x8d;
}
```

FIG. 6

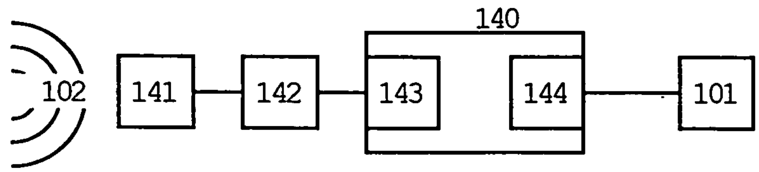

FIG. 7

```
void init(){
 U0BRGR=3333; /* 600 baud if system clock is 32 MHz */
 U0PCR = 1; // Rxd used by UART0
 U0CR1 = UART_EN + RXEN; // enable UART receiver
 U0CR2 = IRTS + WS; // disable hardware RTS control, have 8-bit data frame
 EPDDR = 1; // make port bit 0 an output
}

void check() { short i = 0, j;
 while(U0SR & RRDY) {
  if((j = U0RX) & 0x4000) ;
   else if(j  == 0x8d) i = 1; // output 1 to mute device
   else if(j  == 0xae) timeout = N; // mute for next 3 hours
   else if(j  == 0xe6) timeout = 0; // cancel mute timeout
 }
  if(timeout) i = 0; //
  EPDR = i; // output chosen value
}
```

150 151 152 153 154 155 156 157 158 159 160 161

FIG. 8